TSM Stock Prediction

Student Name: Hsing-Chen (Tony) Lin

Abstract

This project delves into the application of machine learning techniques to predict stock market trends, with a focus on Taiwan Semiconductor Manufacturing (TSM), known for its chipmaking technologies. The study encompasses extensive data preprocessing, incorporating Technical Indicators, Fourier Transform, and ARIMA model predictions to enrich the dataset. The predictive model integrates the strengths of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks. To further elevate its accuracy, the model employs methodologies such as Early Stopping and Bayesian Optimization.

Introduction

- Motivation: The motivation behind this study stems from the intricate and volatile nature of the stock market, influenced by various factors, making it a challenging yet intriguing field for analysis. Engaging in this topic not only immerses me in the domain of machine learning but also provides a valuable chance to gain insights into stock market dynamics and the relationships within its datasets.
- Reason for Choosing TSM: TSM is selected due to its historical pattern of stable growth (2003-2020) coupled with recent shifts towards increased volatility (2020-2024), providing a dataset that offers a unique mix of predictability and unpredictability. This evolution is clearly illustrated in the trend chart.

- Problems:
  - Data Preprocessing – Insufficient Features: the need for extensive data preprocessing to compensate for insufficient features in the dataset, crucial for capturing the complex dynamics of the stock market.
  - Appropriate Model Selection for Time-Series Data: the project requires the careful selection of a suitable machine learning model that can effectively handle time-series data, a characteristic inherent to stock market datasets.
  - Hyperparameter Setting and Overfitting Prevention: Identifying the right balance in hyperparameter values and implementing strategies to avoid overfitting are crucial for the robustness and reliability of the predictive model.

Exploratory Data Analysis

The dataset presented is retrieved from Yahoo Finance, focusing on the stock performance of Taiwan Semiconductor Manufacturing Co. Ltd. (TSM). The dataset covers the period from January 2, 2003, to November 30, 2023, providing a detailed overview of the stock's behavior over two decades.

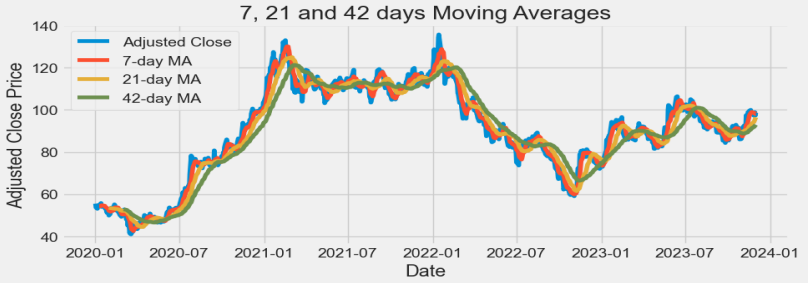
This is the table for the first 5 rows

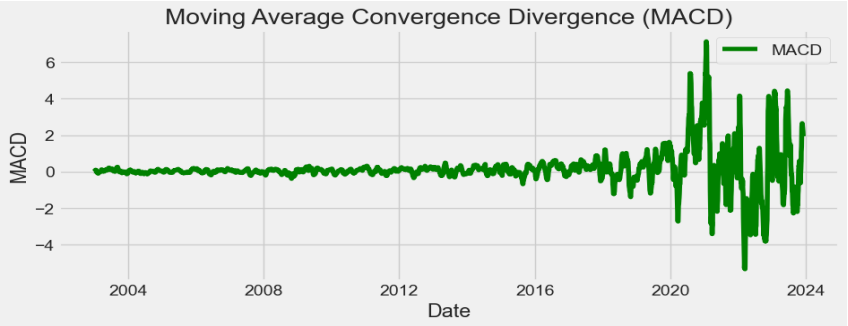
Date	Open	High	Low	Close	Adj Close	Volume
2003-01-02	5.34	5.47	5.26	5.44	2.79	6774506
2003-01-03	5.45	5.61	5.45	5.57	2.85	5766574
2003-01-06	5.59	5.79	5.56	5.72	2.93	7918400
2003-01-07	5.78	5.84	5.68	5.74	2.94	7438789
2003-01-08	5.68	5.69	5.48	5.49	2.82	5619907

1. **Date**: The specific trading day for the stock data as the primary index for the dataset.
2. **Open**:The price at which the stock started trading at the beginning of the trading day.
3. **High**: This indicates the highest price point the stock reached during the trading day.
4. **Low**: The lowest price at which the stock traded during the day.
5. **Close**: The price at which the stock closed at the end of the trading day. It is often used as a standard measure for the stock's performance on a given day.
6. **Adj Close**: The adjusted closing price accounts for any corporate actions such as dividends, stock splits, or new stock issuance.
7. **Volume**: The total number of shares traded during the day.

Apart from that, I will derive technical indicators from the raw dataset to enrich the feature set. These indicators are not readily observable in the initial raw figures. The following are some of the technical indicators I employed in my analysis.

Moving Average: It smoothens price data to create a single flowing line, making it easier to identify the direction of the trend. In my study, I focus on the 7-day, 21-day, and 42-day Moving Averages.





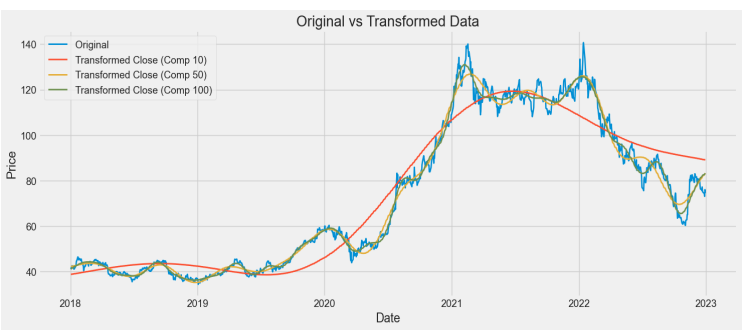
MACD (Moving Average Convergence Divergence): MACD is a trend-following momentum indicator that shows the relationship between two moving averages of a security's price. It can be used to identify potential buy and sell signals.

Methodology

In my training and prediction strategy, I train my model on 90% of the data and test its predictive performance on the remaining 10%. Furthermore, I utilize a sliding window approach where data from the past 60 days is used to forecast the stock price on the 61st day. By continually sliding this window across the dataset, the model is exposed to a diverse range of historical data, enabling it to make informed predictions about future stock prices based on recent trends and movements.

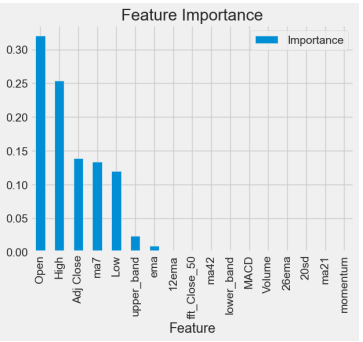
- The preprocessing of the dataset

1. **Technical Indicator Integration**: I augment the dataset with critical technical indicators, such as Moving Averages and Bollinger Bands. These indicators are vital for identifying trends and volatility in the stock market, providing valuable insights that raw price data alone might not reveal.
2. **Fourier Transform**: I apply the Fourier Transform to extract additional features from the 'Close' prices, focusing on trend analysis and noise reduction. This mathematical technique transforms the data into frequencies, allowing me to isolate the components (the parameter).



These components represent the most significant cyclical trends in the stock's behavior. For my analysis, I utilize component 50 to extract data (Orange line) as an additional feature.

3. **ARIMA** Model Predictions: ARIMA is a statistical model used for forecasting time series data. It's effective in understanding and predicting trends in sequential data like stock prices. Therefore, my idea is to utilize ARIMA to forecast future stock prices, and these forecasts are used as an additional feature in the dataset.
4. **Feature Importance Analysis** and **Dimensionality Reduction**:
   I utilize **XGBoost**, a machine learning algorithm based on gradient boosting. The primary use of XGBoost is to determine the most significant features that influence stock price predictions. By identifying these key drivers, I can focus my analysis more effectively on the variables that have the most substantial impact on stock price movements. The right-hand chart displays the importance level of each feature.



   In the dimensionality reduction approach, I use an **Autoencoder**, a specialized neural network, to compress those less critical features of the dataset. In constructing the autoencoder, I built an encoder with dense, batchNormalization, and dropout layers for regularization, progressively reducing the dimension to the specified encoding size. Subsequently, I constructed a decoder to reconstruct it back to its original size.

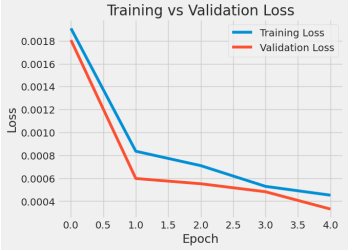
- Training Model

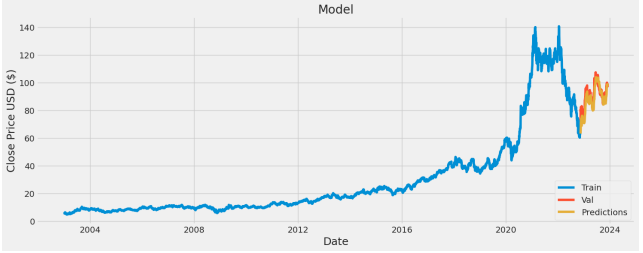
- Combination of Convolutional Neural Networks and Long Short-Term Memory Networks (**CNN-LSTM**): Although CNNs are traditionally adept at image-related tasks, they are particularly effective at pattern recognition and feature extraction. In addition, LSTMs excel at handling sequential data with their ability to preserve information over long periods. Therefore, I suppose that this combination fits training time-series data like stock prices.
- Model Architecture: The design incorporates CNN layers activated by ReLU for efficient feature extraction, followed by LSTM layers for sequential data processing. The model is finalized with dense layers for output.
- **Learning Rate Scheduler**: I implemented a CyclicalSchedule that dynamically adjusts the learning rate within each training cycle, based on a triangular schedule pattern. This schedule helps to avoid local minima and achieve faster convergence.
- **Early Stopping**: I allocated 10% of the training data for validation purposes, which monitors the validation loss and halts training if there's no significant improvement (min_delta = 0.001) after three epochs.
- **Bayesian Optimization**: I set ranges for key hyperparameters like pool size in the MaxPooling2D layers (1 to 5) and epochs (1 to 7). This technique explored within these intervals to identify the optimal combination of parameters, aiming to minimize the model's root mean square error (RMSE).

Results and Comparison

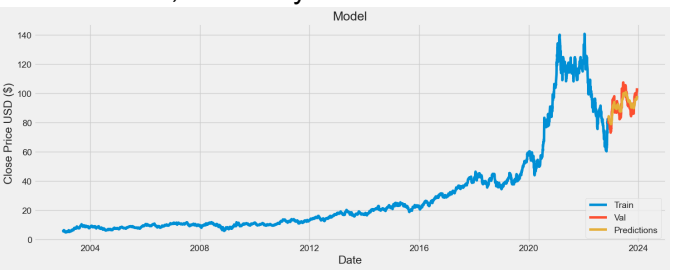
Evaluating Stock Price Prediction Model: RMSE and Accuracy (Predicted values were expected to fall within a range of plus or minus five units from the actual stock prices.)

1. RMSE: 4.51, Accuracy: 0.77



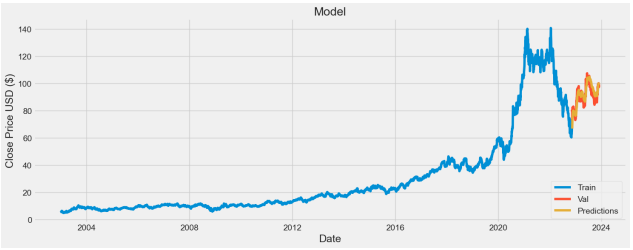


2. Model without Autoencoder: RMSE 3.39, Accuracy 0.89



Discussion: In comparison with the first result, while the autoencoder in the model effectively reduces data complexity and enhances training efficiency, this process potentially sacrifices crucial information. Consequently, the model without the autoencoder, although requiring more time for training, delivered superior accuracy.

3. Model with Autoencoder and using Gru to replace LSTM: RMSE 3.92, Accuracy 0.78



Discussion: In an attempt to streamline the model, I replaced the LSTM with the less complex GRU while retaining the autoencoder. The result indicates that the GRU substitution slightly improved accuracy while reducing the RMSE compared to the model with LSTM, showcasing the effectiveness of GRU in handling the task with reduced complexity.

4. Model with PCA instead of Autoencoder: RMSE 5.05, Accuracy 0.66
   Discussion: In an experimental shift, I replaced the autoencoder with PCA, a linear dimensionality reduction technique. It resulted in a higher RMSE and lower accuracy, likely because PCA's linear approach is less effective at capturing the complex, non-linear patterns and relationships inherent in the data, compared to the more nuanced processing of an autoencoder.

Conclusion

In this project, I utilized an array of techniques to enhance my dataset and refine the predictive model. My approach included integrating Technical Indicators and applying Fourier Transform for data augmentation, along with ARIMA for insightful trend forecasting. After that, I employed XGBoost for feature importance analysis and an Autoencoder for dimensionality reduction, effectively identifying key features and streamlining the dataset for enhanced model efficiency. The combination of CNN and LSTM networks formed the backbone of my model, capitalizing on their strengths in pattern recognition and sequential data handling. By implementing Early Stopping and Bayesian Optimization strategies, I fine-tuned the model to achieve a balance between accuracy and overfitting. Consequently, based on the predictive accuracy of the model, predicting accurate stock prices is still a challenging task. However, I believe that Machine Learning can provide valuable insights and indicators for understanding market movements.

References

1. Initial concept and EDA of the dataset: Stock Market Analysis+Prediction using LSTM
2. ARIMA, Learning Rate Scheduler: Using the latest advancements in AI to predict stock market movements
3. Fourier Transform: Referenced from ChatGPT
4. XGBoost: Using XGBoost in Python Tutorial
5. Autoencoder: Building Autoencoders in Keras
6. CNN-LSTM: Stock Market prediction using CNN-LSTM, Construct a CNN-LSTM model for stock price prediction
7. BayesianOptimization